Network Working Group
Request for Comments: 964

Deepinder P. Sidhu
Thomas P. Blumer
SDC - A Burroughs Company
November 1985

SOME PROBLEMS WITH THE SPECIFICATION OF THE MILITARY STANDARD TRANSMISSION CONTROL PROTOCOL

STATUS OF THIS MEMO

The purpose of this RFC is to provide helpful information on the Military Standard Transmission Control Protocol (MIL-STD-1778) so that one can obtain a reliable implementation of this protocol standard. Distribution of this note is unlimited.

Reprinted from: Proc. Protocol Specification, Testing and Verification IV, (ed.) Y. Yemini, et al, North-Holland (1984).

ABSTRACT

This note points out three errors with the specification of the Military Standard Transmission Control Protocol (MIL-STD-1778, dated August 1983 [MILS83]). These results are based on an initial investigation of this protocol standard. The first problem is that data accompanying a SYN can not be accepted because of errors in the acceptance policy. The second problem is that no retransmission timer is set for a SYN packet, and therefore the SYN will not be retransmitted if it is lost. The third problem is that when the connection has been established, neither entity takes the proper steps to accept incoming data. This note also proposes solutions to these problems.

1. Introduction

In recent years, much progress has been made in creating an integrated set of tools for developing reliable communication protocols. These tools provide assistance in the specification, verification, implementation and testing of protocols. Several protocols have been analyzed and developed using such tools.

In a recent paper, the authors discussed the verification of the connection management of NBS class 4 transport protocol (TP4). The verification was carried out with the help of a software tool we developed [BLUT82] [BLUT83] [SIDD83]. In spite of the very precise specification of this protocol, our analysis discovered several errors in the current specification of NBS TP4. These errors are incompleteness errors in the specification, that is, states where there is no transition for the reception of some input event. Our analysis did not find deadlocks, livelocks or any other problem in the connection management of TP4. In that paper, we proposed

solutions for all errors except for errors associated with 2 states whose satisfactory resolution may require redesigning parts of TP4. Modifications to TP4 specification are currently underway to solve the remaining incompleteness problems with 2 states. It is important to emphasize that we did not find any obvious error in the NBS specification of TP4.

The authors are currently working on the verification of connection management of the Military Standard Transmission Control Protocol (TCP). This analysis will be based on the published specification [MILS83] of TCP dated 12 August 1983.

While studying the MIL standard TCP specification in preparation for our analysis of the connection management features, we have noticed several errors in the specification. As a consequence of these errors, the Transmission Control Protocol (as specified in [MILS83]) will not permit data to be received by TCP entities in SYN_RECVD and ESTAB states.

The proof of this statement follows from the specification of the three-way handshake mechanism of TCP [MILS83] and from a decision table associated with ESTAB state.

2. Transmission Control Protocol

The Transmission Control Protocol (TCP) is a transport level connection-oriented protocol in the DoD protocol hierarchy for use in packet-switched and other networks. Its most important services are reliable transfer and ordered delivery of data over full-duplex and flow-controlled virtual connections. TCP is designed to operate successfully over channels that are inherently unreliable, i.e., they can lose, damage, duplicate, and reorder packets.

TCP is based, in part, on a protocol discussed by Cerf and Kahn [CERV74]. Over the years, DARPA has supported specifications of several versions of this protocol, the last one appeared in [POSJ81]. Some issues in the connection management of this protocol are discussed in [SUNC78].

A few years ago, DCA decided to standardize TCP for use in DoD networks and supported formal specification of this protocol following the design of this protocol discussed in [POSJ81]. A detailed specification of this protocol given in [MILS83] has been adopted as the DoD standard for the Transmission Control Protocol, a reliable connection-oriented transport protocol for DoD networks.

A TCP connection progresses through three phases: opening (or

synchronization), maintenance, and closing. In this note we consider data transfer in the opening and maintenance phases of the connection.

3. Problems with MIL Standard TCP

One basic feature of TCP is the three-way handshake which is used to set up a properly synchronized connection between two remote TCP entities. This mechanism is incorrectly specified in the current specification of TCP. One problem is that data associated with the SYN packet can not be delivered. This results from an incorrect specification of the interaction between the accept_policy action procedure and the record_syn action procedure. Neither of the 2 possible strategies suggested in accept_policy will give the correct result when called from the record_syn procedure, because the recv_next variable is updated in record_syn before the accept_policy procedure is called.

Another problem with the specification of the three-way handshake is apparent in the actions listed for the Active Open event (with or without data) when in the CLOSED state. No retransmission timer is set in these actions, and therefore if the initial SYN is lost, there will be no timer expiration to trigger retransmission. This will prevent connection establishment if the initial SYN packet is lost by the network.

The third problem with the specification is that the actions for receiving data in the ESTAB state are incorrect. The accept action procedure must be called when data is received, so that arriving data may be queued and possibly passed to the user.

A general problem with this specification is that the program language and action table portions of the specification were clearly not checked by any automatic syntax checking process. Several variable and procedure names are misspelled, and the syntax of the action statements is often incorrect. This can be confusing, especially when a procedure name cannot be found in the alphabetized list of procedures because of misspelling.

These are some of the very serious errors that we have discovered with the MIL standard TCP.

4. Detailed Discussion of the Problem

Problem 1: Problem with Receiving Data Accompanying SYN

The following scenario traces the actions of 2 communicating entities during the establishment of a connection. Only the simplest case is considered, i.e., the case where the connection is established by the exchange of 3 segments.

TCP entity A			TCP entity B
state transition	segment recvd or sent by A	segment recvd or sent by B	state transition
			CLOSED -> LISTEN
CLOSED -> SYN_SENT	SYN -->		
		SYN -->	LISTEN -> SYN_RECVD
		<-- SYN ACK	
SYN_SENT -> ESTAB	<-- SYN ACK		
	ACK -->		
		ACK -->	SYN_RECVD -> ESTAB

As shown in the above diagram, 5 state transitions occur and 3 TCP segments are exchanged during the simplest case of the three-way handshake. We now examine in detail the actions of each entity during this exchange. Special attention is given to the sequence numbers carried in each packet and recorded in the state variables of each entity.

In the diagram below, the actions occurring within a procedure are shown indented from the procedure call. The resulting values of sequence number variables are shown in square brackets to the right of each statement. The sequence number variables are shown with the entity name (A or B) as prefix so that the two sets of state variables may be easily distinguished.

Transition 1 (entity B goes from state CLOSED to state LISTEN). The user associated with entity B issues a Passive Open.

```
   Actions: (see p. 104)
      open; (see p. 144)
      new state := LISTEN;
```

Transition 2 (entity A goes from state CLOSED to SYN_SENT). The user associated with entity A issues an Active Open with Data.

```
   Actions: (see p. 104)
      open; (see p. 144)
      gen_syn(WITH_DATA); (see p. 141)
         send_isn := gen_isn();                [A.send_isn = 100]
         send_next := send_isn + 1;           [A.send_next = 101]
         send_una := send_isn;                 [A.send_una = 100]
         seg.seq_num := send_isn;             [seg.seq_num = 100]
         seg.ack_flag := FALSE;          [seg.ack_flag = FALSE]
         seg.wndw := 0;                              [seg.wndw = 0]
         amount := send_policy()              [assume amount > 0]
      new state := SYN_SENT;
```

Transition 3 (Entity B goes from state LISTEN to state SYN_RECVD). Entity B receives the SYN segment accompanying data sent by entity A.

```
   Actions: (see p. 106)
      (since this segment has no RESET, no ACK, does have SYN, and
      we assume reasonable security and precedence parameters, row
      3 of the table applies)
      record_syn; (see p. 147)
         recv_isn := seg.seq_num; [B.recv_isn = seg_seq_num = 100]
         recv_next := recv_isn + 1;            [B.recv_next = 101]
         if seg.ack_flag then
            send_una := seg.ack_num;                  [no change]
         accept_policy; (see p. 131)
            Accept in-order data only:
               Acceptance Test is
                  seg.seq_num = recv_next;
            Accept any data within the receive window:
               Acceptance Test has two parts
                  recv_next =< seg.seq_num =< recv_next +
                                                        recv_wndw
                  or
                  recv_next =< seg.seq_num + length =<
                                                  recv_next + recv_wndw
                  ***********************************************
                  An error occurs here, with either possible
                  strategy given in accept_policy, because
                  recv_next > seg.seq_num.  Therefore
                  accept_policy will incorrectly indicate that
                  the data cannot be accepted.
                  ***********************************************
gen_syn(WITH_ACK); (see p. 141)
   send_isn := gen_isn();                    [B.send_isn = 300]
   send_next := send_isn + 1;               [B.send_next = 301]
   send_una := send_isn;                     [B.send_una = 300]
   seg.seq_num := send_next;                [seg.seq_num = 301]
   seg.ack_flag := TRUE;                  [seg.ack_flag = TRUE]
   seg.ack_num := recv_isn + 1;             [seg.ack_num = 102]
new state := SYN_RECVD;
```

Transition 4 (entity A goes from state SYN_SENT to ESTAB) Entity A receives the SYN ACK sent by entity B.

Actions: (see p. 107)
In order to select the applicable row of the table on p. 107, we first evaluate the decision function ACK_status_test1.

```
ACK_status_test1();
    if(seg.ack_flag = FALSE) then
        return(NONE);
    if(seg.ack_num <= send_una) or
        (seg.ack_num > send_next) then
            return(INVALID)
    else
        return(VALID);

    ... and so on.
```

The important thing to notice in the above scenario is the error that occurs in transition 3, where the wrong value for recv_next leads to the routine record_syn refusing to accept the data.

Problem 2: Problem with Retransmission of SYN Packet

The actions listed for Active Open (with or without data; see p. 103) are calls to the routines open and gen_syn. Neither of these routines (or routines that they call) explicitly sets a retransmission timer. Therefore if the initial SYN is lost there is no timer expiration to trigger retransmission of the SYN. If this happens, the TCP will fail in its attempt to establish the desired connection with a remote TCP.

Note that this differs with the actions specified for transmission of data from the ESTAB state. In that transition the routine dispatch (p. 137) is called first which in turn calls the routine send_new_data (p. 156). One of actions of the last routine is to start a retransmission timer for the newly sent data.

Problem 3: Problem with Receiving Data in TCP ESTAB State

When both entities are in the state ESTAB, and one sends data to the other, an error in the actions of the receiver prohibits the data from being accepted. The following simple scenario illustrates the problem. Here the user associated with entity A issues a Send request, and A sends data to entity B. When B receives the data it replies with an acknowledgment.

TCP entity A			TCP entity B
state transition	segment recvd or sent by A	segment recvd or sent by B	state transition
ESTAB -> ESTAB	DATA -->		
		DATA -->	ESTAB -> ESTAB
		<-- ACK	

Transition 1 (entity A goes from state ESTAB to ESTAB) Entity A sends data packet to entity B.

Actions: (see p. 110)
dispatch; (see p. 137)

Transition 2 (entity B goes from state ESTAB to ESTAB) Entity B receives data packet from entity B.

Actions: (see p. 111)
Assuming the data is in order and valid, we use row 6 of the table.
update; (see p. 159)

****

An error occurs here, because the routine update does nothing to accept the incoming data, or to arrange to pass it on to the user.

****

5. Solutions to Problems

The problem with record_syn and accept_policy can be solved by having record_syn call accept_policy before the variable recv_next is updated.

The problem with gen_syn can be corrected by having gen_syn or open explicitly request the retransmission timer.

The problem with the reception of data in the ESTAB state is apparently caused by the transposition of the action tables on pages 111 and 112. These tables should be interchanged. This solution will also correct a related problem, namely that an entity can never reach the CLOSE_WAIT state from the ESTAB state.

Syntax errors in the action statements and tables could be easily caught by an automatic syntax checker if the document used a more formal description technique. This would be difficult to do for [MILS83] since this document is not based on a formalized description technique [BREM83].

The errors pointed out in this note have been submitted to DCA and will be corrected in the next update of the MIL STD TCP specification.

6. Implementation of MIL Standard TCP

In the discussion above, we pointed out several serious errors in the specification of the Military Standard Transmission Control Protocol [MILS83]. These errors imply that a TCP implementation that faithfully conforms to the Military TCP standard will not be able to

Receive data sent with a SYN packet.

Establish a connection if the initial SYN packet is lost.

Receive data when in the ESTAB state.

It also follows from our discussion that an implementation of MIL Standard TCP [MILS83] must include corrections mentioned above to get a running TCP.

The problems pointed out in this paper with the current specification of the MIL Standard TCP [MILS83] are based on an initial investigation of this protocol standard by the authors.

REFERENCES

[BLUT83] Blumer, T. P., and Sidhu, D. P., "Mechanical Verification and Automatic Implementation of Authentication Protocols for Computer Networks", SDC Burroughs Report (1983), submitted for publication.

[BLUT82] Blumer, T. P., and Tenney, R. L., "A Formal Specification Technique and Implementation Method for Protocols", Computer Networks, Vol. 6, No. 3, July 1982, pp. 201-217.

[BREM83] Breslin, M., Pollack, R. and Sidhu D. P., "Formalization of DoD Protocol Specification Technique", SDC - Burroughs Report 1983.

[CERV74] Cerf, V., and Kahn, R., "A Protocol for Packet Network Interconnection", IEEE Trans. Comm., May 1974.

[MILS83] "Military Standard Transmission Control Protocol", MIL-STD-1778, 12 August 1983.

[POSJ81] Postel, J. (ed.), "DoD Standard Transmission Control Protocol", Defense Advanced Research Projects Agency, Information Processing Techniques Office, RFC-793, September 1981.

[SIDD83] Sidhu, D. P., and Blumer, T. P., "Verification of NBS Class 4 Transport Protocol", SDC Burroughs Report (1983), submitted for publication.

[SUNC78] Sunshine, C., and Dalal, Y., "Connection Management in Transport Protocols", Computer Networks, Vol. 2, pp.454-473 (1978).